Bayesian Anomaly Detection for Ia supernovae using JAX-bandflux

Sam Leeney

With: Harry Bevins, Eloy de Lera Acedo, Will Handley

April 20, 2025

Outline

JAX and JAX-bandflux

Bayesian anomaly detection

Apply on Ia supernovae

JAX and JAX-bandflux

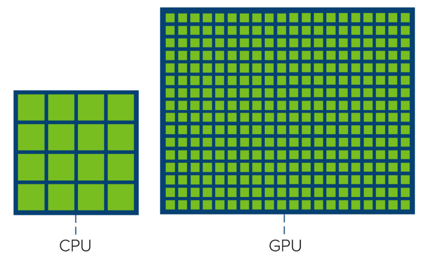
What is JAX

- JAX is a high-performance numerical computing library
- Combines NumPy-like functionality with:
    - Automatic differentiation (like TensorFlow/PyTorch)
    - Just-in-time (JIT) compilation
    - GPU/TPU acceleration
- Fundamentally, allows us to run highly parallelized operations on powerful GPUs

Why do we care about doing things on GPUs

- Massive parallelization capabilities
  - CPUs: Few cores (4-64), high clock speeds
  - GPUs: Thousands of cores, designed for parallel tasks
  - Significant performance improvements for numerical computations

- Useful for astronomical applications
  - Processing large datasets (e.g., supernova surveys)
  - Running complex simulations (e.g., SBI)
  - Performing parameter inference with many samples



Limitations in JAX

- Vectors length must be defined at compilation

Limitations in JAX

- Vectors length must be defined at compilation
- Limited control flow (no while loops, etc)

Limitations in JAX

- Vectors length must be defined at compilation
- Limited control flow (no while loops, etc)
- Debugging and optimising is hard

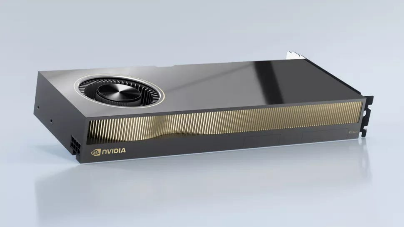
JAX numpy vs numpy

```
import numpy as np
x = np.arange(10)
y = np.zeros_like(x)
for i in range(len(x)):
    y[i] = x[i] * 2
```

```
import jax.numpy as jnp
from jax import vmap
x = jnp.arange(10)
def double(v):
    return v * 2
y = vmap(double)(x)
```

What is JAX bandflux?

sncosmo
A Python Library for Supernova Cosmology



What does JAX-bandflux do?

$$F(p,\lambda) = x_0\big[M_0(p,\lambda) + x_1 M_1(p,\lambda) + \ldots\big] \times \exp\big[c\, CL(\lambda)\big] \tag{1}$$

IN:

- x_0, x_1, t_0, c
- $M_0(p,\lambda), M_1(p,\lambda)$
- p (phase/redshift)
- $CL(\lambda)$

OUT:

- Flux: $F(p,\lambda)$

Bandflux Computation

- Bandflux: Integrated flux through a specific filter

$$\text{bandflux} = \int_{\lambda_{\text{min}}}^{\lambda_{\text{max}}} F(\lambda) \cdot T(\lambda) \cdot \frac{\lambda}{hc} \, d\lambda \tag{2}$$

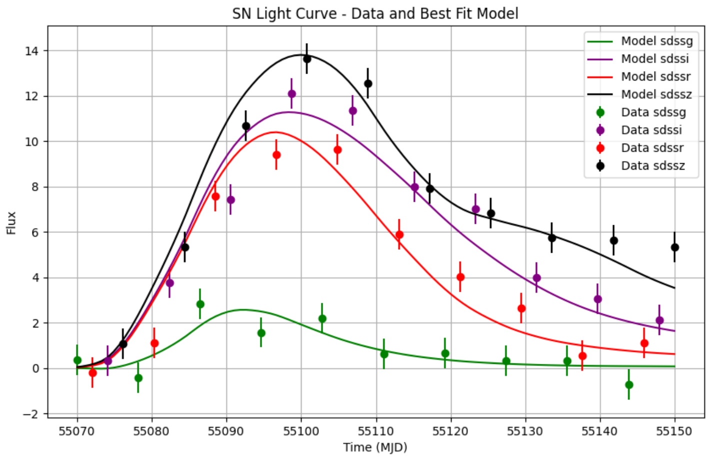
- Components:
  - $F(\lambda)$: Spectral flux density
  - $T(\lambda)$: Filter transmission function
  - $\lambda/(hc)$: Conversion from energy to photon counts
- Implementation in JAX-bandflux:
  - Vectorized array operations
  - JIT-compiled for efficiency
  - Trapezoidal integration along wavelength dimension
  - Parallelized across multiple data instances via `vmap`

Can now construct GPU compatible likelihood

$$\mathcal{L}(\mathbf{F} \mid \theta) = \prod_i \frac{1}{\sqrt{2\pi}\,\sigma_i} \exp\left[-\frac{(F_i - \mu_i(\theta))^2}{2\,\sigma_i^2} \right]$$

- Observed fluxes F_i: measured data; model fluxes $\mu_i(\theta)$: predicted flux given parameters
- With this likelihood we can now perform parameter estimation
- Likelihood is differentiable and can be compiled as part of broader GPU workflows

Can now fit light curves on GPU

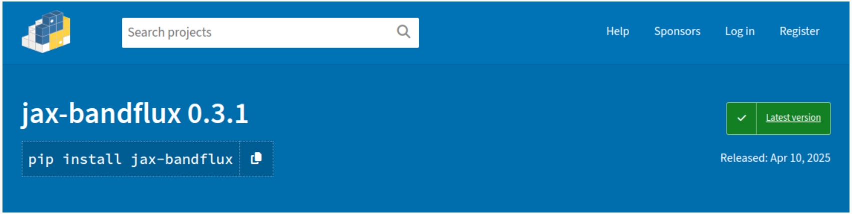


Bandflux Computation Example

- For a detailed walkthrough of bandflux computation:
  - Interactive Jupyter notebook with step-by-step implementation
  - Demonstrates JAX-bandflux API usage
  - Shows performance benefits of GPU acceleration
  - Includes visualization of results
- Access the example notebook:

Supernovae Analysis Example Notebook

PIP installable



JAX-bandflux: differentiable supernovae SALT modelling for cosmological analysis on GPUs

Samuel Alan Kossoff Leeney[1, 2]

1 Astrophysics Group, Cavendish Laboratory, J. J. Thomson Avenue, Cambridge CB3 0HE, UK **2** Kavli Institute for Cosmology, Madingley Road, Cambridge CB3 0HA, UK

Bayesian anomaly detection

Over simplified example of anomaly detection method (thresholding)

Simple statistical approach:

- Calculate mean (μ) and standard deviation (σ) of the data
- Define threshold $T = \mu + k\sigma$ where k is a sensitivity parameter
- Flag data point x_i as anomalous if:

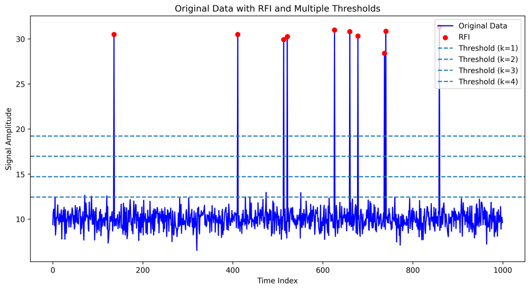
$$\text{anomaly}_i = \begin{cases} 1 & \text{if } x_i > T \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Limitations:

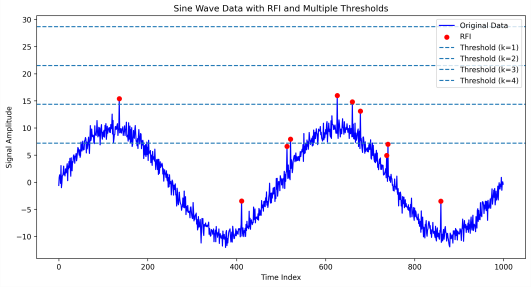
- Choice of k is arbitrary
- Assumes Gaussian statistics
- No consideration of temporal correlations
- Cannot distinguish between RFI and real signals

Thresholding results

Constant signal with RFI:



Sine wave with RFI:



Defining an anomaly sensitive likelihood

a) Generate peicewise likelihood:

$$P(\mathcal{D}_i|\theta) = \begin{cases} \mathcal{L}_i(\theta) & : \text{expected} \\ \Delta^{-1}[0 < \mathcal{D}_i < \Delta] & : \text{anomalous}, \end{cases} \tag{4}$$

b) Ascribe Bernoulli prior:

$$P(\varepsilon_i) = p_i^{(1-\varepsilon_i)}(1-p_i)^{\varepsilon_i}. \tag{5}$$

c) Marginalise over epsilon:

$$P(\mathcal{D}|\theta) = \sum_{\varepsilon \in \{0,1\}^N} P(\mathcal{D}, \varepsilon|\theta) \tag{6}$$

d) Approximate correct mask is most likely

$$P(\mathcal{D}|\theta, \varepsilon_{\max}) \gg \max_j P(\mathcal{D}|\theta, \varepsilon^{(j)}), \tag{7}$$

e) Loglikelihood:

$$\log P(\mathcal{D}|\theta) = \sum_i [\log \mathcal{L}_i + \log(1-p_i)]\varepsilon_i^{\max} + [\log p_i - \log \Delta](1-\varepsilon_i^{\max}) \tag{8}$$

Computing the Posterior

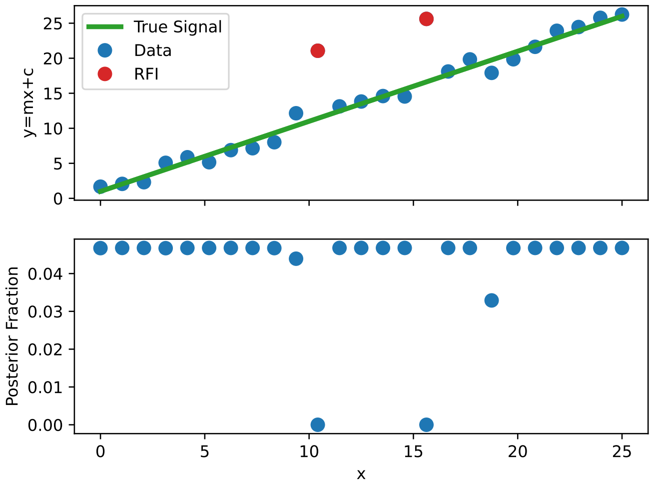
e) Loglikelihood:

$$\log P(\mathcal{D}|\theta) = \sum_i [\log \mathcal{L}_i + \log(1 - p_i)]\varepsilon_i^{\mathrm{max}} + [\log p_i - \log \Delta](1 - \varepsilon_i^{\mathrm{max}}) \tag{6}$$

f) Maximise ε^{max} by comparing the terms:

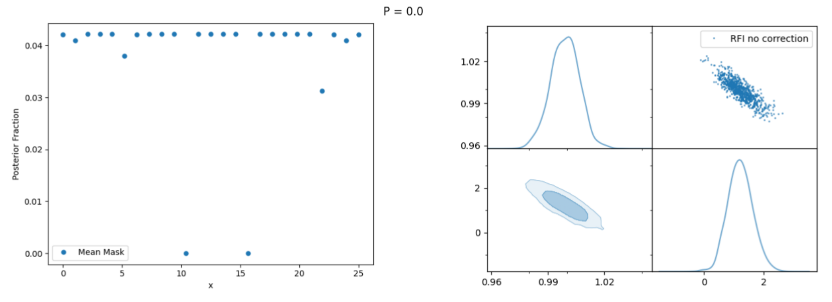
$$\log P(\mathcal{D}|\theta) = \begin{cases} \log \mathcal{L}_i + \log(1 - p_i), & \text{if } [\log \mathcal{L}_i + \log(1 - p_i) > \log p_i - \log \Delta] \\ \log p_i - \log \Delta, & \text{otherwise} \end{cases} \tag{7}$$

Fit on a simple toy model

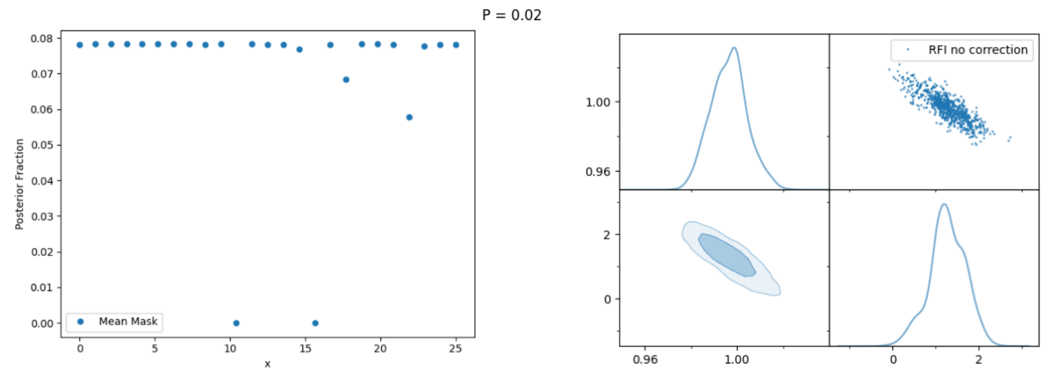


Likelihood thresholding condition p

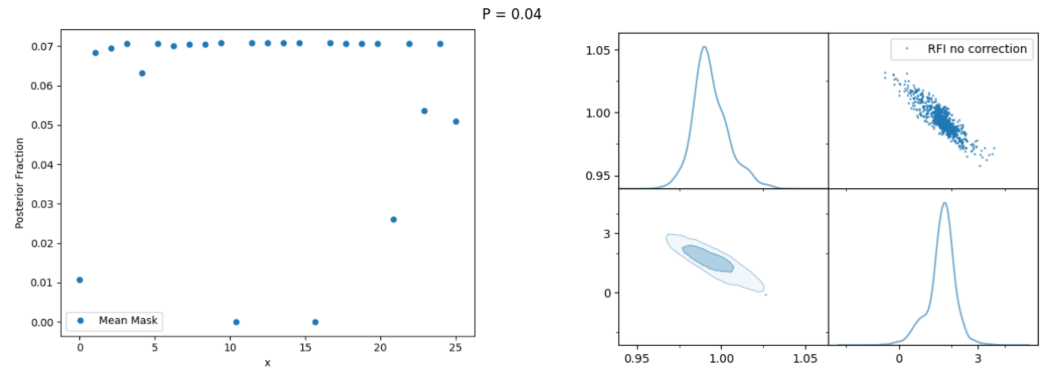
$$\log P(\mathcal{D}|\theta) = \sum_i [\log \mathcal{L}_i + \log(1-p_i)]\varepsilon^{\max} + [\log p_i - \log \Delta](1-\varepsilon_i^{\max}) \tag{9}$$



Varying p


P = 0.02
Posterior Fraction
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Mean Mask
0
5
10
15
20
25
x
RFI no correction
1.00
0.96
2
0
0.96
1.00
0
2

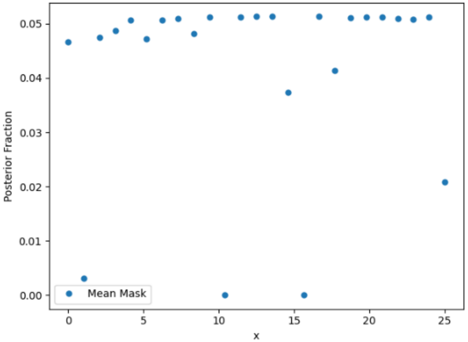
Varying p

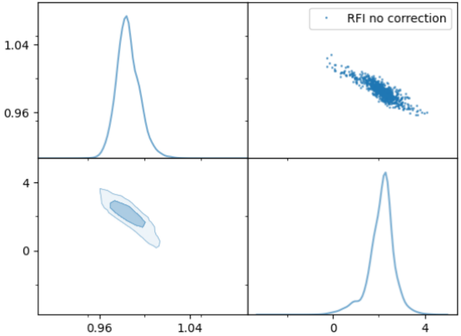


P = 0.04
Posterior Fraction
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Mean Mask
0
5
10
15
20
25
x
RFI no correction
1.05
1.00
0.95
3
0
0.95
1.00
1.05
0
3

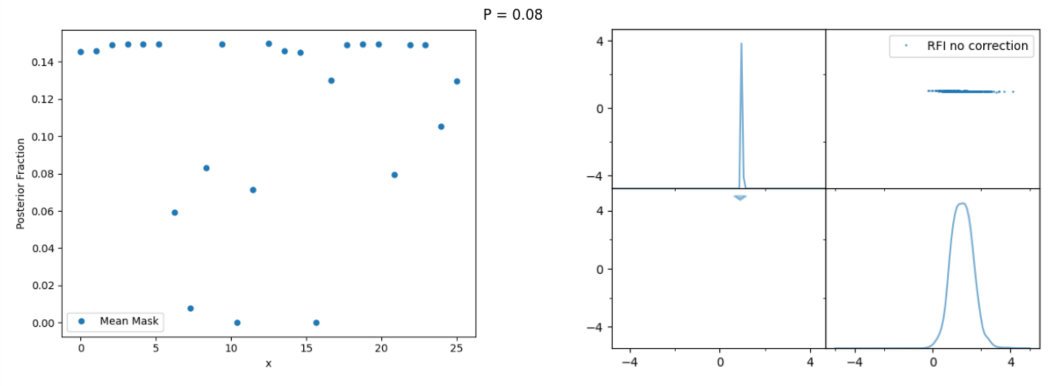
Varying p

P = 0.06



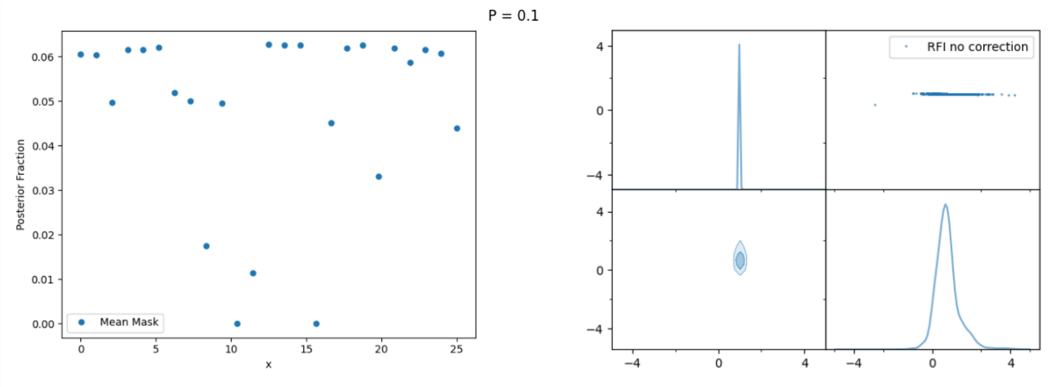


Varying p



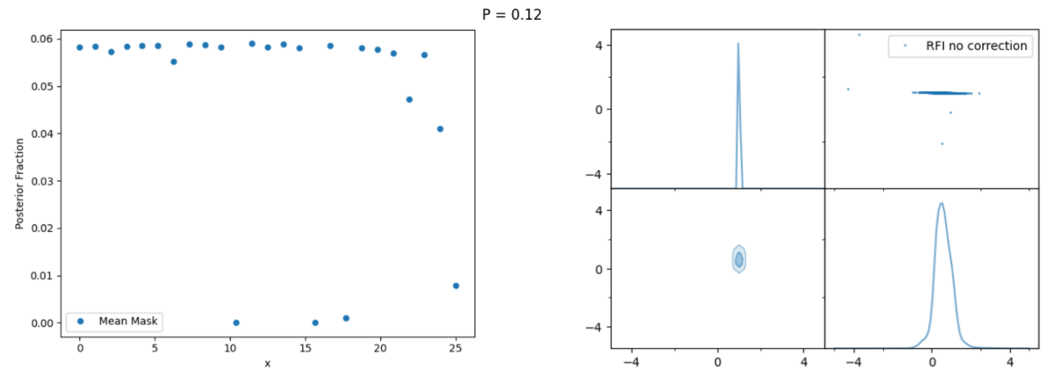
P = 0.08
Posterior Fraction
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
Mean Mask
0
5
10
15
20
25
x
RFI no correction
4
0
−4
4
0
−4
−4
0
4
−4
0
4

Varying p

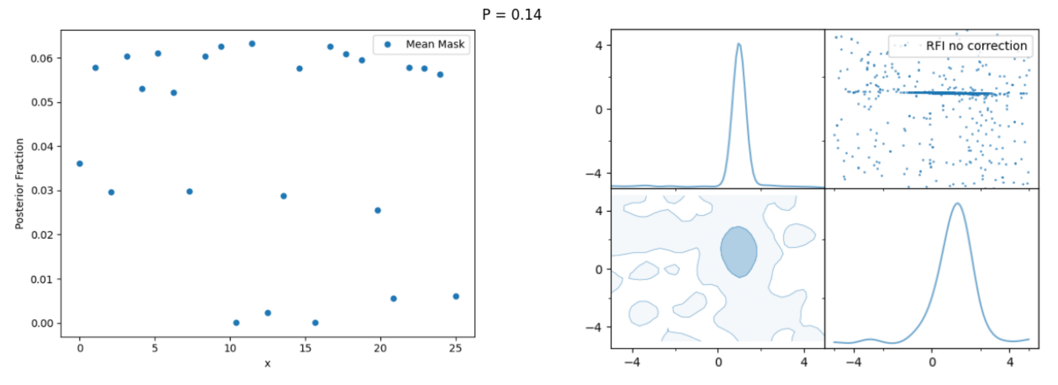


P = 0.1
Posterior Fraction
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Mean Mask
0
5
10
15
20
25
x
RFI no correction
4
0
−4
4
0
−4
−4
0
4
−4
0
4

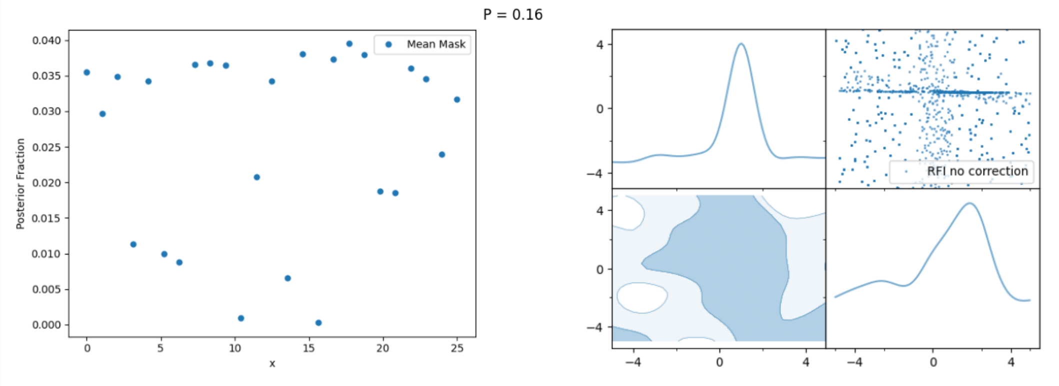
Varying p


P = 0.12
Posterior Fraction
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Mean Mask
0
5
10
15
20
25
x
RFI no correction
4
0
−4

Varying p

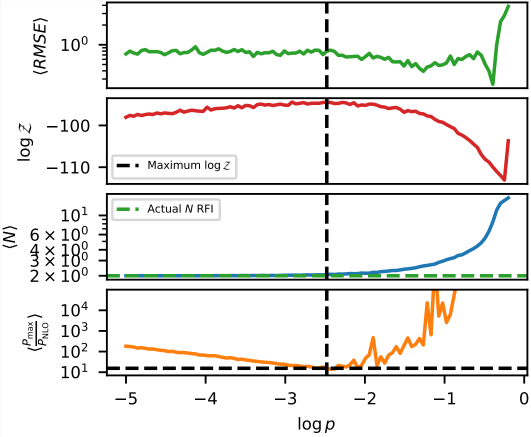

P = 0.14
Mean Mask
Posterior Fraction
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
5
10
15
20
25
x
RFI no correction
4
0
−4
4
0
−4
−4
0
4
−4
0
4

Varying p


P = 0.16
Posterior Fraction
Mean Mask
x
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
5
10
15
20
25
RFI no correction
4
0
−4

Selection strategy for p.

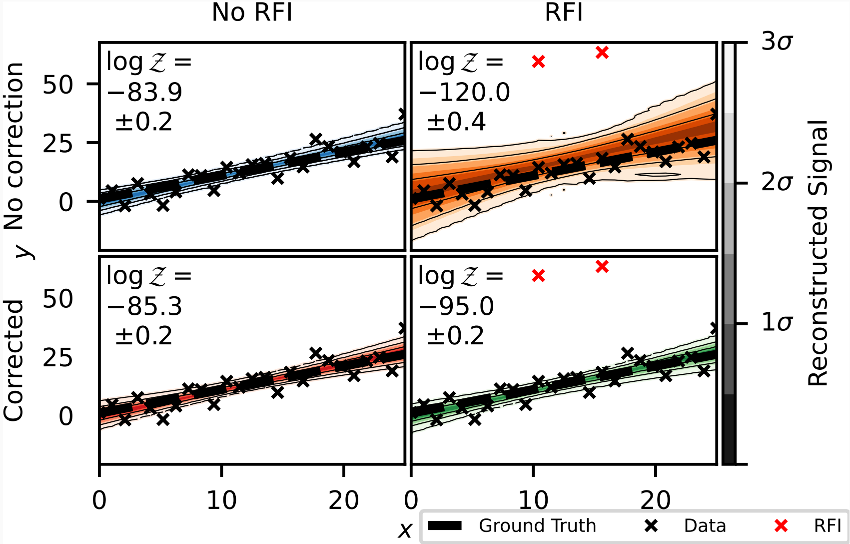
- 'Select p such that the Bayesian evidence is maximised'



Fully automated anomaly detection

- Putting a prior on p, we can fit it dynamically as a free parameter.
- This fully automates the anomaly detection process.
- Must exclude $p = 0$.

Application to toy model



Implement with 2 lines of code

```python
def likelihood(theta):
    sig = theta[0]
    logL = -(f_noise - window)**2/sig**2/2 - np.log(2*np.pi*sig**2)/2
    return logL, []
```

```python
def likelihood(theta):
    sig = theta[0]
    logL = -(f_noise - window)**2/sig**2/2 - np.log(2*np.pi*sig**2)/2 + np.log(1-p)
    emax = logL > logp - np.log(delta)
    logPmax = np.where(emax, logL, logp - np.log(delta)).sum()
    return logPmax, []
```

Tutorial @ github.com/samleeney

Read the paper!

PHYSICAL REVIEW D **108**, 062006 (2023)

Bayesian approach to radio frequency interference mitigation

S. A. K. Leeney,[*,§] W. J. Handley,[†,§] and E. de Lera Acedo[‡,§]
The University of Cambridge, Astrophysics Group, Cavendish Laboratory, J. J. Thomson Avenue, Cambridge, CB3 0HE, United Kingdom

(Received 5 May 2023; accepted 29 August 2023; published 29 September 2023)

Interfering signals such as radio frequency interference from ubiquitous satellite constellations are becoming an endemic problem in fields involving physical observations of the electromagnetic spectrum. To address this we propose a novel data cleaning methodology. Contamination is simultaneously flagged and managed at the likelihood level. It is modeled in a Bayesian fashion through a piecewise likelihood that is constrained by a Bernoulli prior distribution. The techniques described in this paper can be implemented with just a few lines of code.

DOI: 10.1103/PhysRevD.108.062006

arxiv: 2211.15448

Apply on Ia supernovae

Standard vs. Anomaly Detection Likelihoods

Standard Likelihood:

$$\log \mathcal{L}_{\text{std}} = -\frac{1}{2}\sum_i \left(\frac{f_i - m_i}{\sigma_i}\right)^2 - \frac{1}{2}\sum_i \log(2\pi\sigma_i^2) \qquad (10)$$

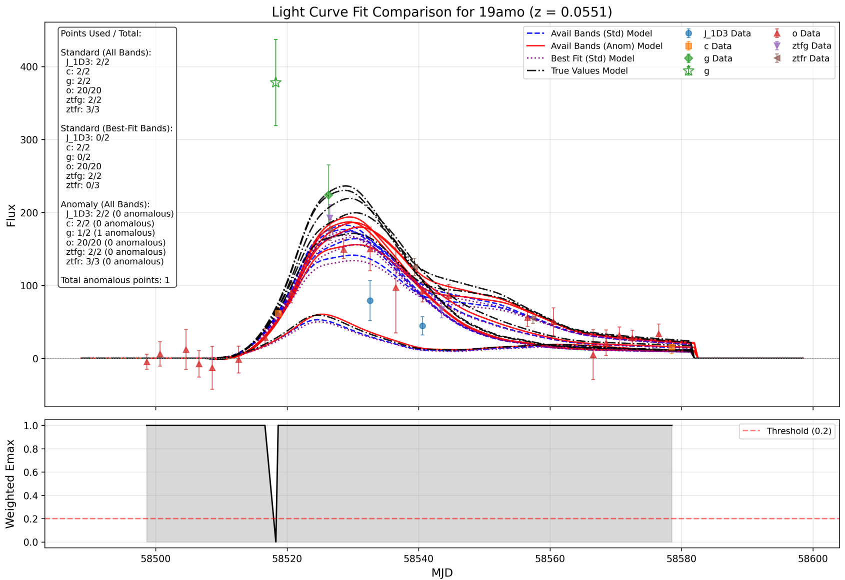
- f_i: Observed flux
- m_i: Model flux (SALT3)
- σ_i: Flux uncertainty

Anomaly Detection Likelihood:

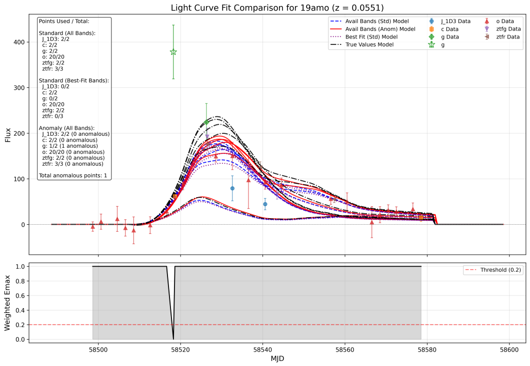
$$\log \mathcal{L}_{\text{anom}} = \sum_i \begin{cases} \log \mathcal{L}_i + \log(1-p), & \text{if } e_i^{\max} \\ \log p - \log \Delta, & \text{otherwi} \end{cases} \qquad (11)$$

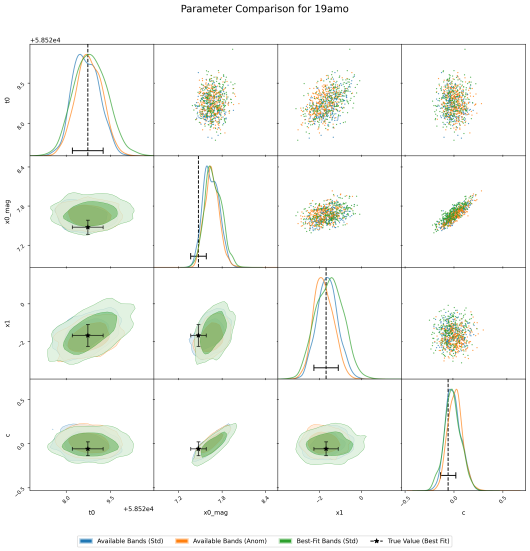
- $\log \mathcal{L}_i$: Point-wise standard likelihood
- p: Anomaly probability (fitted parameter)
- $e_i^{\max}$: Boolean indicating normal data
- Δ: Maximum flux range

Applying to Ia supernovae

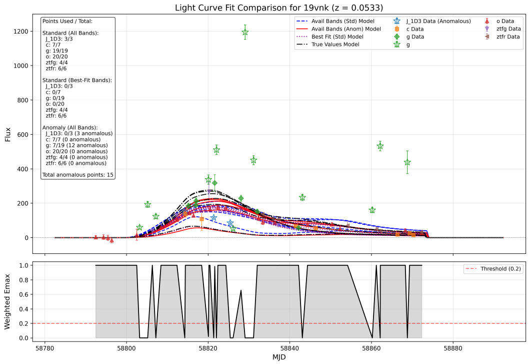


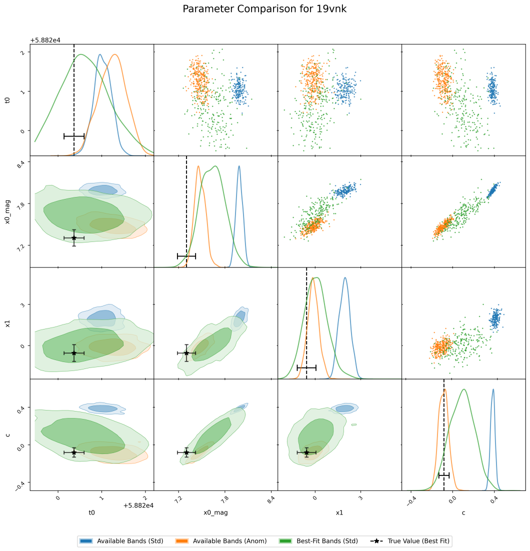
SN 19amo: Classic 'anomaly detection' example



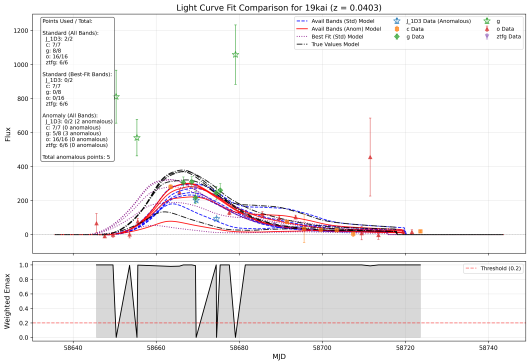


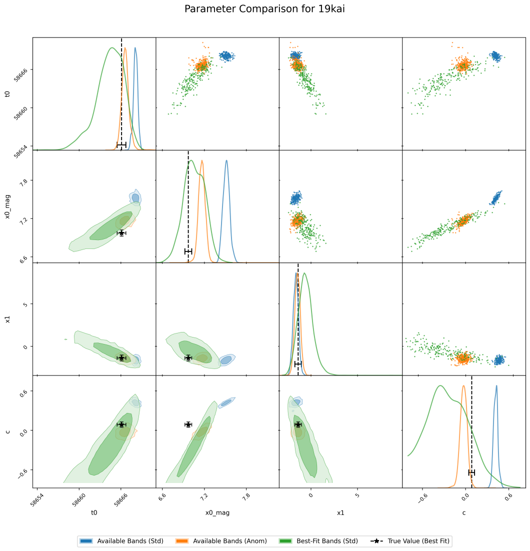
SN 19vnk: Automatic filter removal



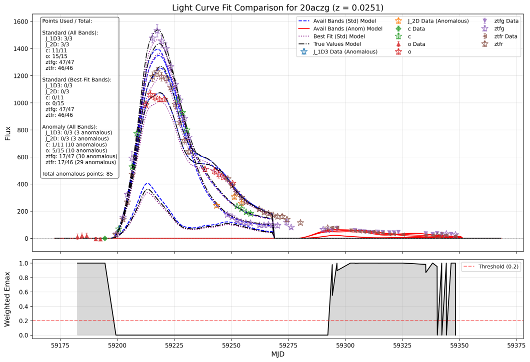


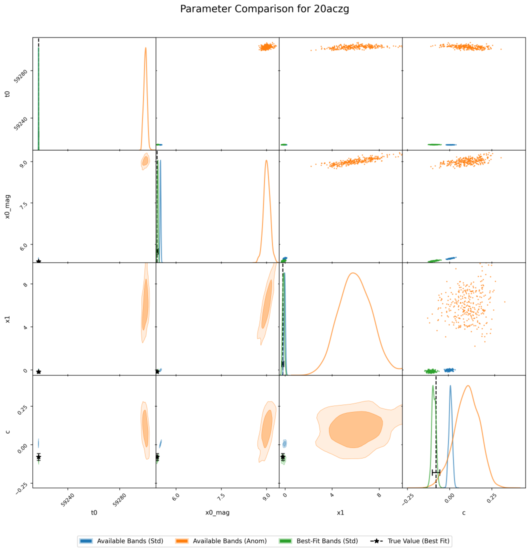
SN 19kai: Flagging while preserving some data





SN 20aczg: Light Curve and Corner Plot Comparison





Key points

1. Standard flagging.

Key points

1. Standard flagging.
2. Automated filter selection.

Key points

1. Standard flagging.
2. Automated filter selection.
3. Data preservation from previously discarded filters.
4. Potentially can flag non Ia automatically?

Next steps

- Assess Hubble diagrams
    - Quantify impact on cosmological parameter estimation
    - Compare with traditional outlier rejection methods
    - Evaluate systematic error reduction
- Try on other datasets?
    - Apply to different supernova surveys (ZTF, LSST)
    - Test with different photometric systems
    - Evaluate performance across redshift ranges